Validation of ATL Transformation to Generate a Reliable MVC2 Web Models

M'hamed RAHMOUNI
Department of Computer Science, Faculty of Science
Ibn Tofail University
Kenitra, BP 133, Morocco

Samir MBARKI
Department of Computer Science, Faculty of Science
Ibn Tofail University
Kenitra, BP 133, Morocco

Abstract—**Technologies are constantly evolving. In order to benefit from technological advances, it is necessary to adapt the applications to these technologies. This operation is expensive for companies because it is often necessary to rewrite the code entirely. Where there is no capitalization of application functions and development is generally based on source code, the separation of concerns appears to be the necessary solution to the problem. Thus, functional specifications and technical specifications are taken into account separately by MDA approach. In this paper we present a new method of transformation validation and then we implement a new model transformation process based on MDA approach to generate an MVC2 Web model from Struts 2. This transformation begins by the validation of different transformation rules by applying the developed method of transformation validation.**

Keywords- MDA; Validation of transformation; Struts; ATL transformation; MVC2 architecture.

I. Introduction

The key requirement for development methodologies have always been motivated by the complexity and indeterminism of software engineering. In despite the multitude of approaches, very few achieve unanimity in the community. Most of them are adequate for a particular application field, and are generally based on a set of contextual beliefs and assumptions. In particular, object-oriented methodologies attempt to visualize, model and implement software as a set of interacting objects. The enthusiasm aroused by the object paradigm is such that dozens of methodologies have emerged since the nineties, making the choice of one method rather than another difficult. In response to this proliferation, in 2000, OMG launched the MDA approach, which is based on the concerns separation. It makes it possible to take into account, separately, the business aspect and the technical aspect of an application, thanks to the modeling. The application source code is obtained by automatic generation from the application models. Models are no longer just a visual or communication element, but are, in MDA approach [6]-[7], a productive element and a pivot of MDA process. In addition to MDA approach, companies are oriented towards using frameworks such as Struts2 [1], Hibernate [2] and Spring [3] [4].

The most methodologies of Web systems development [5], are as well based on model-driven engineering approach. Thus approaches of Web Engineering such as: WebML [8], OO-H [9], OOWS [10], UWE [11] and WebSA [12] propose to build different Web systems views following a horizontal concerns separation.

This work allows generates automatically an MVC2 web model that is a PSM model. This latter respects the architecture of MVC2 pattern. To arrive to this objective, we prepare PIM and PSM meta-models then we establish the different traceability links between these meta-models. After establishing the different traceability links, we applied our method of transformation validation which we will present in the next sections to ensure that our transformation rules are correct and valid. Then thereafter we define the different rules by ATL transformation language. Finally, we explain our method of transformation validation by applying these rules with a case study.

This paper is structured as follows: section 2 presents the process and methodology of this work. Section 3 explains MDA approach. Section 4 describes the validating method. Section 5 is devoted to the architecture of UML and Struts2 meta-models. Section 6 presents the transformation rules implementation. The transformation rules execution and the result of the execution process is the subject of section 7. Section 8 discusses the main related work, while section 9 wraps up the conclusions and future works.

II. Process and Methodology

In this work, the process starts by the presentation of a new method of transformation validation then the meta-modeling of Struts 2 framework allows implement the different CIM, PIM, and PSM meta-models. The CIM model of this work is an UML class diagram of a case study of an Employee management. The functional model (PIM) is a simplified UML meta-model. The PSM meta-model is a Struts2 meta-model. The PSM meta-model is presented in the figure 3. The next step is to define the ATL transformation rules and validation of these rules by the developed method. After this, we begin by the implementation of KM3 models corresponding to each meta-model then the different Ecore models corresponding to each KM3. The last step is to establish the traceability links between the components of source and target meta-models then, we define the different transformation rules in ATL transformation language. The result of this work is the MVC2 web model represented in EMF model. This is the configuration file of the proposed application. From the generated PSM model, we can generate the application code of the case study by applied an M2C transformation. The M2C transformation is neglected in this work. It will be the subject of future work.

The tools support of this work is the UML, ATL transformation language, MOF, XMI, KM3, OCL and EMF Project.

III. MDA (Model-Driven Architecture)

After procedural technology, object technology and component technology, the MDA [13] (Model-Driven Architecture) approach is a Model-Driven Engineering (MDE) process. MDA is proposed by the OMG (Object Management Group) in 2000. This approach is based on the separation of concerns. It allows take into account, separately, business aspect and technical aspect of an application, thanks to the modeling. The source code of application is obtained by automatic generation from application models. In MDA approach, the models are no longer just a visual or communication element, but are a productive element and a pivot of MDA process. To achieve a modeling or transformation process, MDA uses multiple standards such as UML [14], MOF [15], XMI [16], OCL [17] and many others.

This work is depicted to develop a validating method of transformation and the test of this method by a transformation example. In the following section, we present the theoretical and practical framework of new validation method.

IV. Validating Method

Model transformations, especially the transformation rules specification constitute a major problem in context of model-driven engineering, which requires a great deal of precision, analysis and testing to arrive at the suitable transformations rules. In most cases of transformations, one falls into the ambiguity which obliges us to reconstruct our rules again, then to test if the code is the one we seek. Testing and rebuilding the rules is one of the time-wasting factors that all organizations are trying to minimize as much as possible. This operation constitutes a primordial phase in the case of transformations model. To solve this problem and ensure our rules before starting the transformation phase (translation of rules into code), as well as to keep the traces of these rules, we propose a new method of transformations validation, which will be called: MVT method, which is based essentially on a Petri Net and UML notations. In this paper, we present a new method of transformations validation. To explain this method we begin by a theoretical study as a principle of this method then the application of this method by using a case study example. The principle of this method is the subject of the following section.

A. The Method principle

In IDM context, a transformation corresponds to a T function between two modeling languages M and N [13]. The T transformation can be decomposed into two or more transformations t_1, t_2, t_3, ...,t_n such that $T = t_1.t_2 \ldots t_n$; "." Is the composition relation and T:: MMS→MMC (MMS: Source MetaModel and MMC: Target Metamodel).

The meta-classes of source and target meta-models are represented in this approach by the places of Petri Net whose name is the same name that refers to the place.

Each transformation rule is represented by a transition t_i. The composition relation between target metamodel elements and which results from t_i rule is represented by a C_{ij} transition. This later is an inner rule.

The T function represents the global transformation which makes it possible to obtain the target metamodel in its entirety from the source meta-model elements.

According to the method principle, we can conclude that: $T=\prod_{i=1}^{n}\left(t_i\bigcup_{j=1}^{m}C_{ij}\right)$ where t_i is a transformation rule number i, which makes it possible to transform an element of source metamodel to one or more elements of target metamodel. The C_{ij} relation is an inner transformation rule, transformation between elements of target metamodel, which means that a class is composed of a set of elements of another class. The compound element may be a resultant element of same transformation t_i or a resulting element of another transformation t_j. The first relation of composition will be called: simple composition noted C_{ii}. While the second relation of composition, it will be called: complex composition and it is noted C_{ij}.

If we put $t_k = t_i\bigcup_{j=1}^{m}C_{ij}$ and $T = \prod_{k=1}^{n}t_k$ then we get:

$T = t_1.t_2 \ldots\ldots\ldots .t_n$.

B. Application of Validation Method

Figure 1 shows the transformation rules validation, in accordance with the above cited method (Figure 1), we can ensure that these rules can generate all elements of target meta-model that they will need. There are the elements of target metamodel represented by the MVT method. This method is represented in the form of a flowchart which corresponds between the elements of source and target metamodel. This flowchart is represented in the following figure 1:

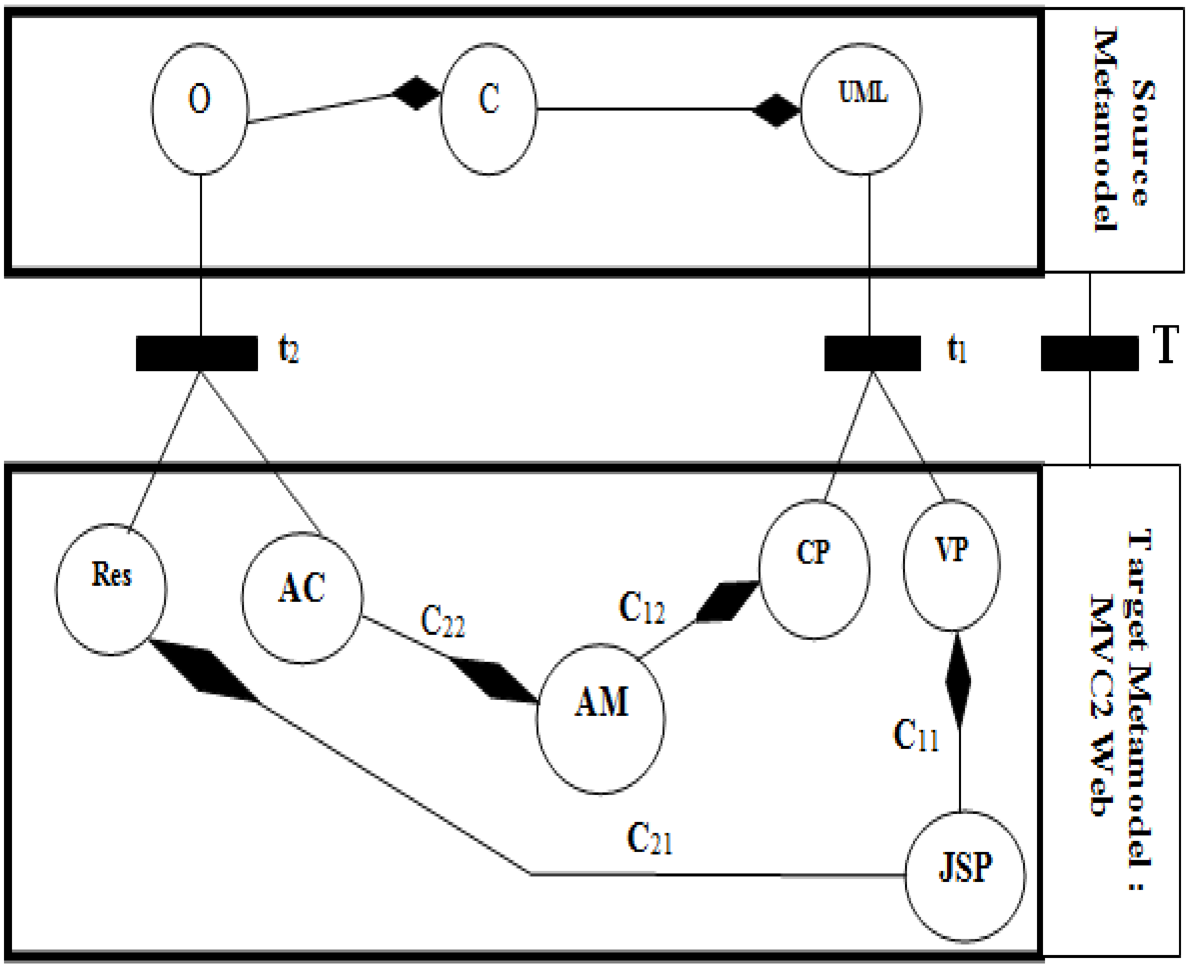


Figure 1. Representation of formation rules by flowchart of MVT method.

Table I. Elements abbreviation of source and target meta-model.

Elements of Source Metamodel	
UML	Package UML
C	Class
O	Operation
Elements of Target Metamodel	
VP	View Package
CP	Controller Package
AM	Action Mapper
JSP	JSP Page
AC	Action
Res	Result

To experiment this method, we conduct an ATL transformation based on MDA approach. In this transformation, we begin by establishing the different transformation rules between elements of source and target metamodel and thereafter we implement and define different rules by ATL language. In this transformation we begin by the definition of each element of source and target metamodel.

The source and target metamodel is the subject of the following section.

V. UML AND STRUTS META-MODELS

In this section, we present the various meta-classes forming the UML source and target meta-models.

A. UML Source Meta-model

The source meta-model structures the simplified UML model based on the package containing the data types and classes. Figure 2 presents the UML source meta-model. The different components of this meta-model are as follow:

- **UmlPackage:** Represents the concept of UML package. This meta-class is linked to the classifier meta-class.
- **Classify:** This is a generalization of meta-classes representing both the concept of UML classes and the concept of data type.
- **Class:** Represents the concept of UML classes.
- **DataType:** Represents the UML data types.
- **Operation:** Expresses the concept of methods of an UML class.

- **Parameter:** Represents the concept of method parameters.
- **Property:** Expresses the concept of properties of an UML class.

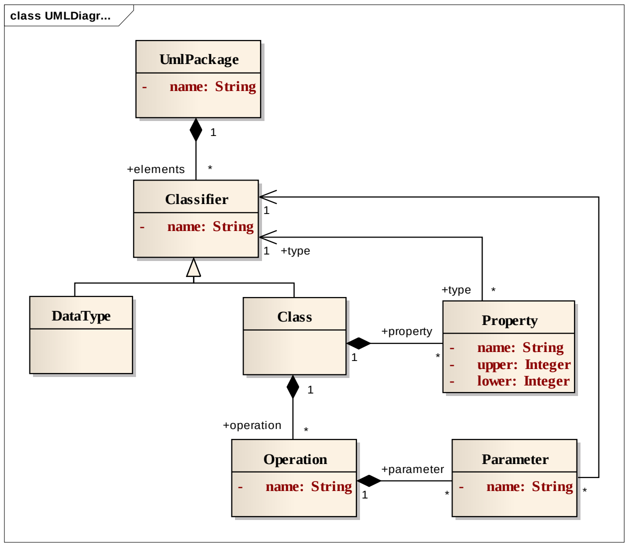


Figure 2. Simplified UML meta-model.

B. *Struts 2 Target Meta-model*

Figure 3 illustrates the target meta-model. This meta-model represents the concept of MVC2 web model. The Struts 2 meta-model is presented in first time in [36]. The different meta-classes of Struts 2 meta-model are as follows:

- **ModelPackage**: Expresses the UML package concept and designed the notion of Model in the MVC2 architecture.
- **ControllerPackage**: Indicates the controller concept in the MVC2 architecture.
- **ViewPackage**: Represents the concept of Views package .
- **ActionMapper**: Expresses the concept of ActionMapper class.
- **ActionProxy**: This is the concept of ActionProxy class.
- **ActionInvocation**: Indicates the concept of ActionInvocation class.
- **Action**: Represents the action concept in the controller package.
- **JspPages**: Indicates the concept of Jsp package.
- **Result**: Represents the concept of the generated result through an Action class.
- **The Interceptors**: This is an Interceptor package.
- **Interceptor**: Expresses the concept of interceptor classes.
- **HttpRequest**: Represents the concept of HttpServletRequest classes.
- **HttpResponse**: Expresses the HttpServletResponse classes concept.
- **Result**: Indicates the concept of Result classes.

VI. TRANSFORMATION RULES IMPLEMENTATION

In this section we present the different steps from implementation to execution of different transformation rules. The first step is to implement the following meta-models: *N-tiers.km3*, *N-tiers.ecore*, *UML.km3*, *UML.ecore*. The second step is to establish the rules specification. After that, we define the different rules based on the specification rules. These rules are written in ATL language in a file named *UML2N-tiers.atl*. Finally, we prepare the source model. This model is an UML class diagram of Employee management. We translate the source model in XMI language.

To achieve this work, we have used different tools like: ATL plug-in integrated in Eclipse, OCL, XMI, UML, EMF project, KM3 and MOF.

In the following sections, we present the ATL transformation language then the rules specification and finally the implementation and execution of ATL transformation rules. The *km3* and *Ecore* meta-models cited above are not presented in this paper for letting it quite understandable and clear.

A. *ATL: Atlas Transformation Language*

ATLAS Transformation Language (ATL) is a model transformation language inspired by the OMG standard QVT. It developed in the framework of ATLAS project at LINA in Nantes [22]-[23]-[24]. ATL is part of Eclipse M2M (Model-to-Model) project [25]. Figure 4 shows the ATL operational framework.

B. *Rules Specification*

In this section, we present the main rules to transform an UML Class Diagram into an N-tiers Web model. The specification rules are as follow:

- The View package is composed of a set of JSP pages.
- Each UML package can generate a Struts2 package.
- The Struts 2 package is composed of a Struts package.
- The Struts package is composed of a Controller package and a View package.
- The Controller Package is composed of a set of Action classes.
- Each Action class is composed of a set of Result classes.
- Each Result is composed of a set of JSP pages.
- Each Operation can generate an Action and a JSP pages.

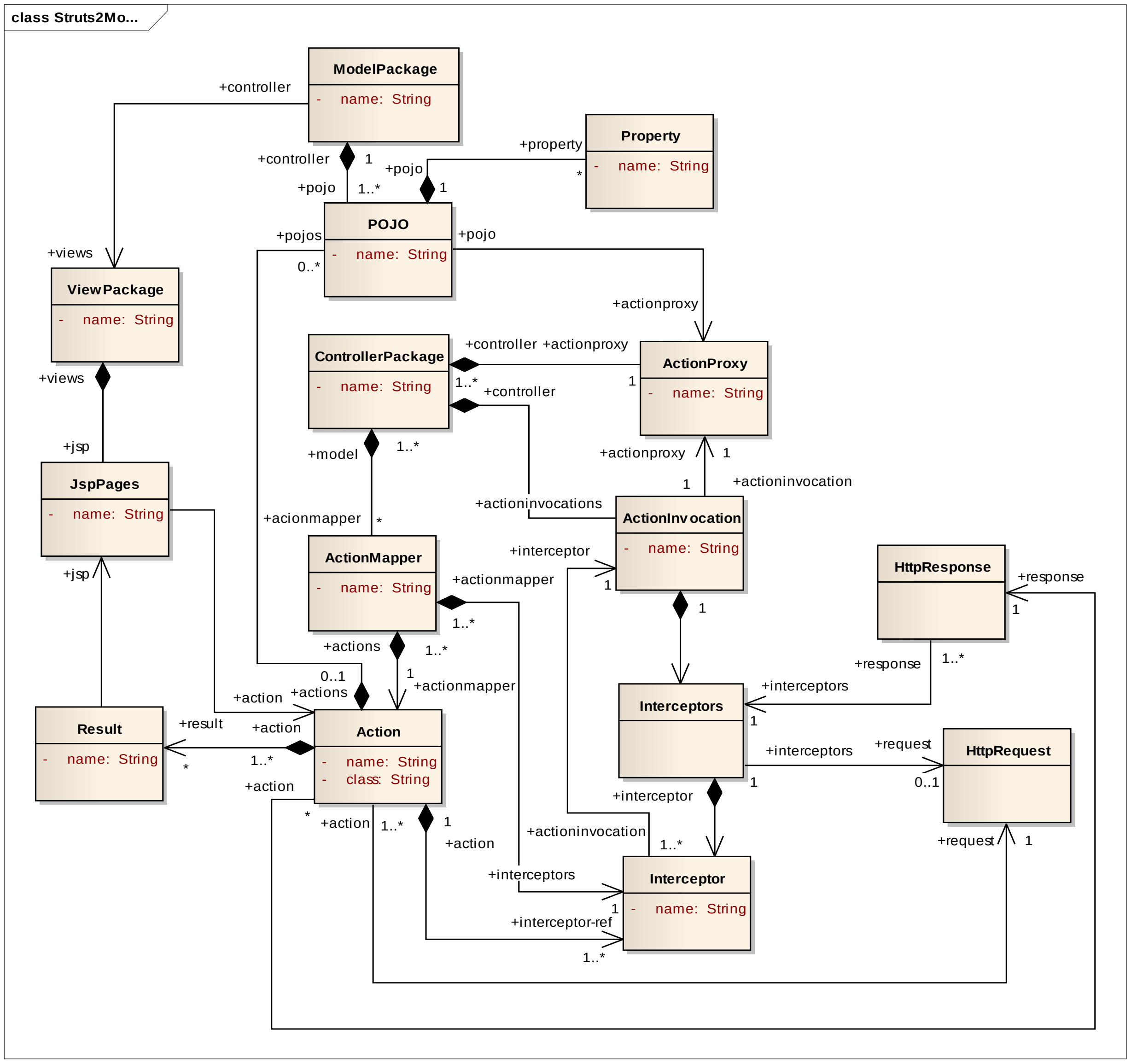


Figure 3. Struts2 Meta-model.

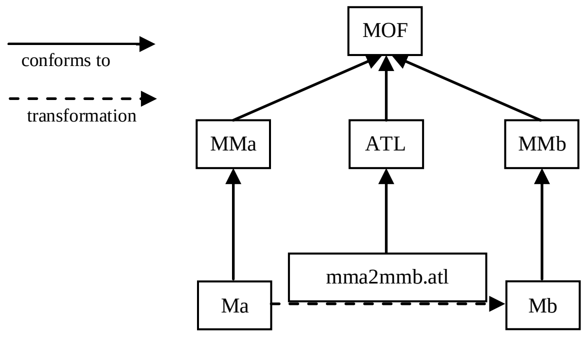


Figure 4. ATL Operational framework.

C. *Rules-Based transformation written in ATL*

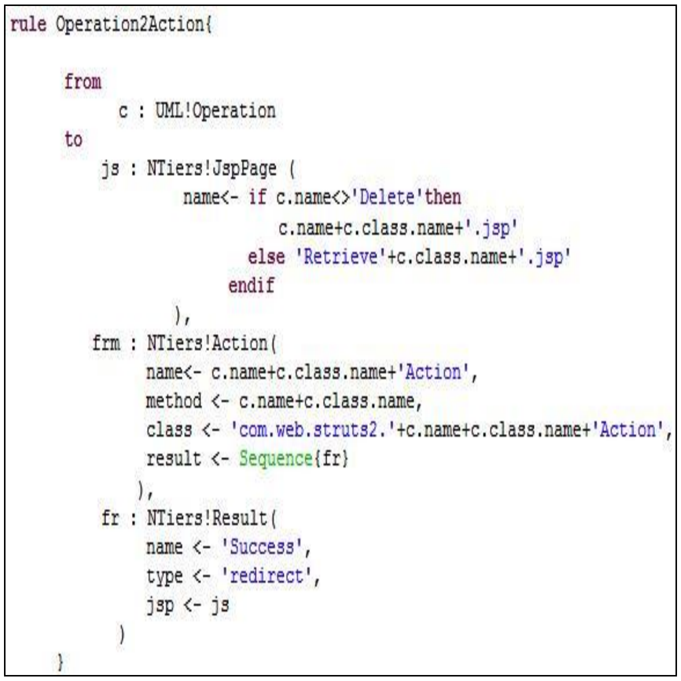
In this section we present the different rules which transform the UML model into MVC2 web model. These rules are as follow:

Main Rule: From Operation to Struts 2 Action

This rule permits to generate the different action classes and jsp result of each Action. In this rule the name of jsp page is the name of the operation concatenated with the name of the class and followed by the extension ".jsp". This rule is composed of a three rules. These rules are as follow:

- Rule 1: From Operation to Action.
- Rule 2: From Operation to Result.
- Rule 3: Each Result is composed of a set of Jsp pages.

The main rule is shown in figure 5. These rules are implemented by ATL language.



```
rule Operation2Action{

     from
          c : UML!Operation
     to
         js : NTiers!JspPage (
                 name<- if c.name<>'Delete'then
                          c.name+c.class.name+'.jsp'
                       else 'Retrieve'+c.class.name+'.jsp'
                     endif
                 ),
        frm : NTiers!Action(
             name<- c.name+c.class.name+'Action',
             method <- c.name+c.class.name,
             class <- 'com.web.struts2.'+c.name+c.class.name+'Action',
             result <- Sequence{fr}
            ),
         fr : NTiers!Result(
             name <- 'Success',
             type <- 'redirect',
             jsp <- js
           )
     }
```

Figure 5. Rule that generates Action classes and Jsp Pages

VII. Transformation Rules Execution

The execution algorithm of ATL transformation allows browsing all transformation rules and thereafter generates the MVC2 web model. This latter is represented in figure 7.

A. Case Study

In this case study, we consider a system of a three classes. This system can manage the employee of a given department. The system classes are: the City class, the Department class and the Employee class. In this system, we use only the CRUD (Create, Retrieve, Update, and Delete) operations. These are most often implemented in all systems. In this case we can generate the model which can manage the requested layer from the layers already defined. Figure 6 shows the UML class diagram of this system

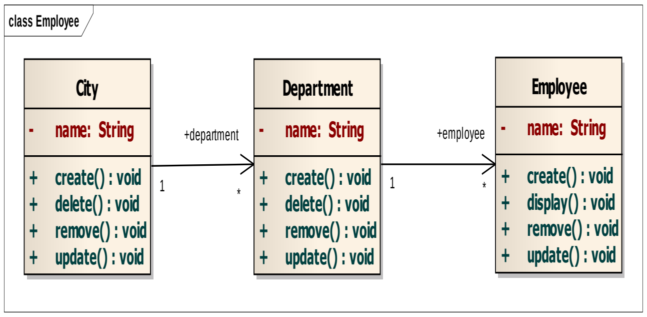


Figure 6. UML class diagram of the employee management system

B. ATL Transformation Result

The generated PSM model respects the web system architecture based on the patterns integration. Indeed, this model is composed of a set of Controller package, View package. The controller package is composed of a set of Action classes. The view package represents the different jsp pages.

Figure 7 shows the generated MVC2 Web model. This model contains the different ingredients for implementing a presentation layer respecting the architecture of MVC2 pattern.

In this section, we present a case study to demonstrate and exemplify our proposition. The UML class diagram of this case study represents the source model of our ATL transformation.

VIII. Related Work

In the last decade, several studies in model transformation and code generation have been conducted. The most relevant are: [26]-[11]-[27]-[28]-[29]-[30]-[31]-[32]-[33]-[34]-[36].

The work [26] allows generate JSP pages and JavaBeans by applying the UWE [11] combined with ATL transformation language [24]. The integration of AJAX into the UWE engineering process is considered as a future work of the authors.

The objective of the work [27] is to transform the PIMs models defined by Enterprise Distributed Object Computing into generated PSMs for different services platforms. The different transformation rules in this paper are defined by ATL language.

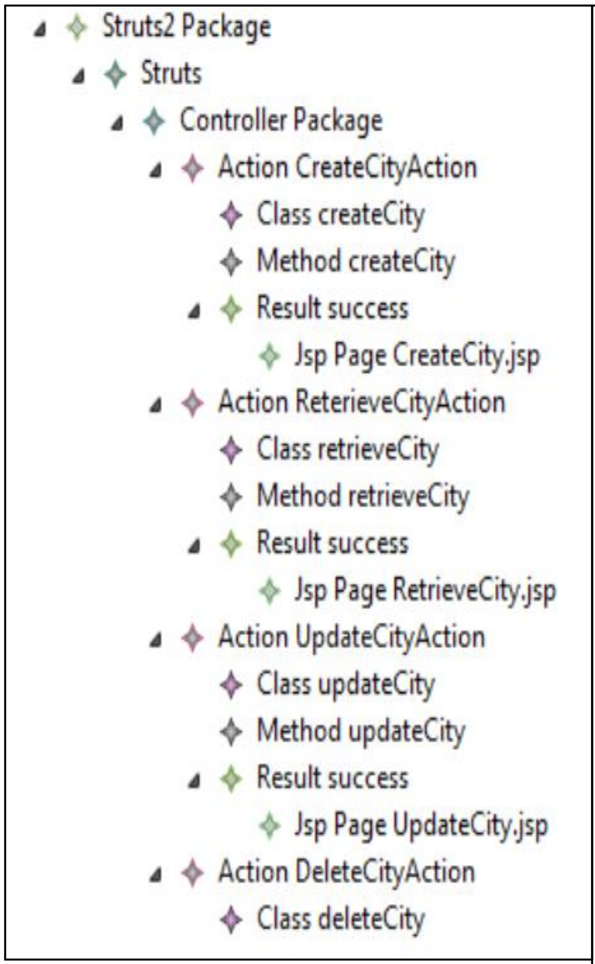
In [28], Billing et al. describes the different transformation rules permits to translate PIM to PSM in the EJB context. This transformation is realized by applying the approach by modeling based on QVT.

The work presented in [29] considers that MDA is a software industrialization pattern (or a software factory). The idea of this work is illustrated by a real case study in an IT services company. The main objective is to create MDA tools founded on XMI, XSLT and Visitor pattern. .It is a proposal to create MDA tools taking as base XMI, XSLT and the Visitor pattern.

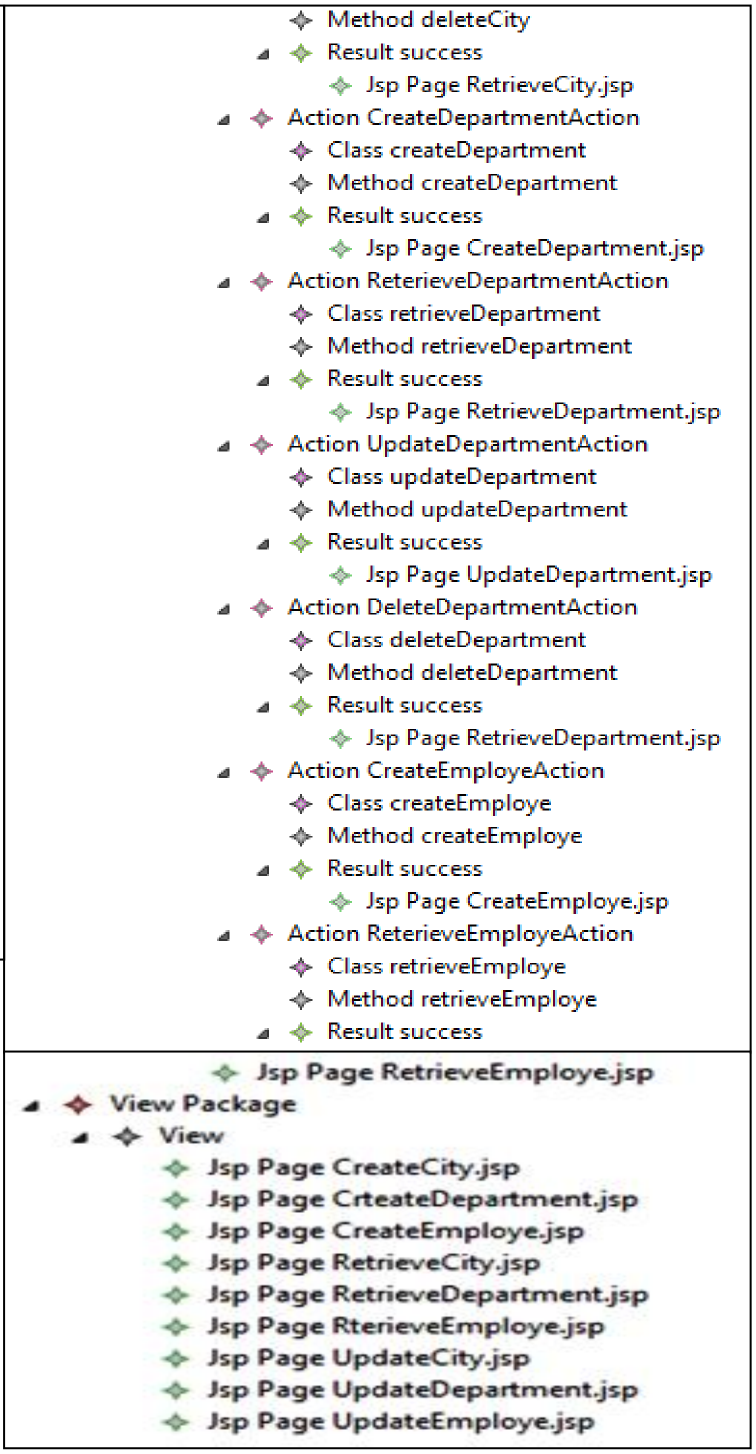
The model-driven development approach for E-Learning platform is the subject of the work [30]. In this work the authors realize the CIM model by analyzing business logic. Then they establish the system diagram and the robustness analysis. Finally, the authors define a transformation method from PIM to PSM layer by layer.

The objective of the work [31] is to integrate a new framework for secure Data Warehouses design by applying MDA approach based on QVT.

The AndroMDA approach has earned attention in the community of web-based MDA [32]. This work allows transform a PIM schemes to model by integrating a wide variety of scenarios and comes with a set of plug-ins, called cartridge.



```
Struts2 Package
  Struts
    Controller Package
      Action CreateCityAction
        Class createCity
        Method createCity
        Result success
          Jsp Page CreateCity.jsp
      Action ReterieveCityAction
        Class retrieveCity
        Method retrieveCity
        Result success
          Jsp Page RetrieveCity.jsp
      Action UpdateCityAction
        Class updateCity
        Method updateCity
        Result success
          Jsp Page UpdateCity.jsp
      Action DeleteCityAction
        Class deleteCity
```



```
        Method deleteCity
        Result success
          Jsp Page RetrieveCity.jsp
      Action CreateDepartmentAction
        Class createDepartment
        Method createDepartment
        Result success
          Jsp Page CreateDepartment.jsp
      Action ReterieveDepartmentAction
        Class retrieveDepartment
        Method retrieveDepartment
        Result success
          Jsp Page RetrieveDepartment.jsp
      Action UpdateDepartmentAction
        Class updateDepartment
        Method updateDepartment
        Result success
          Jsp Page UpdateDepartment.jsp
      Action DeleteDepartmentAction
        Class deleteDepartment
        Method deleteDepartment
        Result success
          Jsp Page RetrieveDepartment.jsp
      Action CreateEmployeAction
        Class createEmploye
        Method createEmploye
        Result success
          Jsp Page CreateEmploye.jsp
      Action ReterieveEmployeAction
        Class retrieveEmploye
        Method retrieveEmploye
        Result success
          Jsp Page RetrieveEmploye.jsp
    View Package
      View
        Jsp Page CreateCity.jsp
        Jsp Page CrteateDepartment.jsp
        Jsp Page CreateEmploye.jsp
        Jsp Page RetrieveCity.jsp
        Jsp Page RetrieveDepartment.jsp
        Jsp Page RterieveEmploye.jsp
        Jsp Page UpdateCity.jsp
        Jsp Page UpdateDepartment.jsp
        Jsp Page UpdateEmploye.jsp
```

Figure 7. The Generated MVC2 Web model.

In [33] the authors arrive to generate the MVC 2 web model from the Struts framework. They define the different transformation rules by ATL language in view to generate the CRUD operations from three classes Ci, Cj and Ck.

The work presented in [34] can generate the MVC 2 web model from the combination of the UML class diagram and the UML activity diagram. The main idea of this combination is to stabilize the UML class diagram and to determine the input jsp page of each Action class. In [35], the authors generate the MVC2 web applications code through the model already generated in [34] by using JET2 template integrated in Eclipse project.

The objective of the paper [36] is to generate the N-tiers web model from the integration of Struts2, Spring IoC and Hibernate DAO frameworks.

Finally, the objective of this paper is to validate the ATL transformation rules presented in this work which was not possible in [33]-[34]-[35]-[36]. This paper describes a new validation method of ATL transformation rules and the application this method.

IX. Conclusion and Future Work

The work presented in this paper is part of a context in which the size and complexity of software increases while the constraints of time, development and quality then the maintenance and evolution are always stronger. To respond to this trend, model engineering appears as a promising evolution of software engineering techniques. However, the success of a development approach is conditioned by the existence of techniques to ensure the quality of the product software. As we have shown in this paper, using models in a productive way requires well-formalized modeling environments and techniques to validate model transformation programs.

Our work is a step in the direction of reliable model engineering and open many perspectives in this field. In this case, we present a new method of transformation validation then we have applied the approach by modeling based on ATL transformation language to generate MVC2 web model from UML class diagram. This transformation is began by defining the traceability links between the UML source meta-model and MVC2 target meta-model already obtained. The algorithm execution of ATL transformations allow browsing all transformation rules and generate MVC2 PSM model respecting the architecture of MVC2 pattern. The generated MVC2 PSM model is an EMF model. This file can be used to produce automatically the necessary target application code. Finally, the transformation result was demonstrated and exemplified by a case study.

Furthermore, we plan to generate an e-business web code from the generated MVC2 web model by applying the model-to-code (M2C) transformations. In other hand, we can extend this method for considering other frameworks like: PHP, Zend and DotNet.

References

[1] Strsus2 homepage (2017). Retrieved February 4, 2017 from http: //www.struts.apache.org/2.x/.

[2] Hibernate homepage. Retrieved February 4, 2017 from http: //hibernate.org/.

[3] Spring homepage (2017). Retrieved February 2, 2017 from http: //projects.spring.io/spring-framework/.

[4] Arnaud Cogoluègnes, Thierry Templier, Julien Dubois , & Jean-Philippe Retaillé (2nd Ed.) . "Spring par la Pratique". 2009, Eyrolles.

[5] GertiKappel, Birgit Pröll, Siegfried Reich, & Werner Retschizegger. "Web Engineering". 2006, John Wiley.

[6] Blanc, X., "MDA en action : Ingénierie logicielle guidée par les modèles ". Eyrolles, 2005.

[7] Kleppe, A., Warmer, J., & Bast, W., "MDA Explained: The Model Driven Architecture: Practice and Promise". 2003, Addison-Wesley Professional.

[8] Stefano Ceri, Piero Fraternali, Aldo Bongio, Marco Brambilla, Sara Comai, & Maristella Matera. "Designing Data-Intensive Web Applications". 2003, Morgan Kaufman.

[9] Jaime Gómez, & Cristina Cachero. "OO-H: Extending UML to Model Web Interfaces". Information Modeling for Internet Applications. 2002. IGI Publishing.

[10] Pedro Valderas, Joan Fons, & Vicente Pelechano. "From Web Requirements to Navigational Design – A Transformational Approach". Proceedings of the 5th Int. Conference of Web Engineering (ICWE'05), 2005, LNCS 3579.

[11] Koch, N., "Transformations Techniques in the Model-Driven Development Process of UWE". Proceeding of the 2nd Workshopsh of Model-Driven Web Engineering (MDWE'06). 2006, Palo Alto.

[12] Santiago Meliá, & Jaime Gomez. "The WebSA Approach: Applying Model Driven Engineering to Web Applications". Journal of Web Engineering, Vol.5, No.2, 2006.

[13] Object Management Group (OMG). Model Driven Architecture. Document update/2012-06-27, Retrieved January 12, 2017 from http://www.omg.org/mda/.

[14] OMG. (2017, January). Unified Modeling Language. Available at : http://www.uml.org/

[15] Meta Object Facility (MOF), version 2.0, Retrieved January 20, 2017 from http://www.omg.org/spec/MOF/2.0/PDF/.

[16] XML Metadata Interchange (XMI), version 2.1.1. Retrieved January 28, 2017 from http://www.omg.org/spec/XMI/.

[17] Object Management Group (OMG) (2017). UML 2 Object Constraint Language (OCL). Retrieved January 30, 2017 from http://www.omg.org/cgibin/doc?ptc/2005-06-06.

[18] Pan Yuqing, & Wang Jiuli. "Design And Implementation of Accounting E-Business Platform for Source Documents". Proceeding in the International Conference on Computer Application and System Modeling (ICCASM 2010). 2010, DOI: 10.1109/ICCASM.2010.5622916.

[19] Spring AOP homepage. Retrieved January 8, 2017 from http://static.springsource.org/spring/docs/2.5.x/reference /aop.html.

[20] Praveen Gupta, & Govil, Prof. M.C., "Spring Web MVC Framework for rapid open source J2EE application development: a case study". International Journal of Engineering Science and Technology, Vol.2, No.6, 2010, pp.1684-1689.

[21] Praveen Gupta, & Govil, Prof. M.C., "MVC Design Pattern for the multi framework distributed applications using XML, spring and struts framework". International Journal on Computer Science and Engineering, Vol.2, No.4, 2010, pp.1047-1051.

[22] MENET, L., "Formalisation d'une Approche d'Ingénierie Dirigée par les Modèles Appliquée au Domaine de la Gestion des Données de Référence". PhD thesis, Université de Paris VIII, Laboratoire d'Informatique Avancée de Saint-Denis (LIASD), école doctorale : Cognition Langage Interaction (CLI), 2010

[23] Frédéric Jouault, & Ivan Kurtev, "Transforming Models with ATL". Proceeding in MoDELS 2005 Workshops, LNCS 3844, 2006, pp. 128 – 138, © Springer-Verlag Berlin Heidelberg.

[24] Jouault, F., Allilaire, F., Bézivin, J., & Kurtev, I., "ATL: A Model Transformation Tool". Sciences of Computer Programming-Elseiver, Vol.72, No.1-2, 2008, pp.31–39.

[25] AtlanMod (2017). Atl transformation language home page. Retrieved January 20, 2017 from http://www.eclipse.org/m2m/atl/.

[26] Kraus, A., Knapp, & A., Koch N., "Model-Driven Generation of Web Applications in UWE". Proceeding in the 3rd International Workshop on Model-Driven Web Engineering, CEUR-WS, Vol. 261, 2007.

[27] Bezivin, J., Hammoudi, S., Lopes, D., & Jouault, F., "Applying MDA approach for web service platform". Proceedings of the 8th IEEE International Entreprise Distributed Object Computing Conference (EDOC'04), 2004, pp.58-70.

[28] Bezivin, J., Busse, S., Leicher, A., & Suss, J.G., "Platform Independent Model Transformation Based on TRIPLE". In Middleware'04: Proceedings of the 5th ACM/IFIP/USENIX International Conference on Middleware, 2004, pp. 493- 51.

[29] Ndie, T. D., Tangha, C., & Ekwoge, F. E., "MDA (Model- Driven Architecture) as a Software Industrialization Pattern: An Approach for a Pragmatic Software Factories". Journal of Software Engineering & Applications, Vol.3, No.6, 2010, pp. 561-571.

[30] Cong, X., Zhang, H., Zhou, D., Lu, P., & Qin, L., "A Model- Driven Architecture Approach for Developing E-Learning Platform", Entertainment for Education. Proceeding of Digital Techniques and Systems Lecture Notes in Computer Science, Vol. 6249/2010, 2010, pp. 111-122, DOI: 10.1007/978-3-642-14533-9-12.

[31] Soler, E., Trujillo, J., Blanco, C., & Fernandez-Medina, E., "Designing Secure Data Warehouses by Using MDA and QVT". Journal of Universal Computer Science, Vol.15, No.8, 2009, pp.1607-1641.

[32] AndroMDA (2017). Retrieved January 15, 2017 from http: //www.andromda.org/.

[33] RAHMOUNI, M., & Mbarki, S., "MDA-Based ATL Transformation to Generate MVC 2 Web Models". International Journal of Computer Science & Information Technology (IJCSIT), Vol.3, No.4, 2011, pp. 57-70.

[34] RAHMOUNI, M., & Mbarki, S., "Combining UML Class and Activity Diagrams for MDA Generation of MVC 2 Web Applications". International Review in Computers and Software (IRECOS), Vol.8, No.4, 2013, pp.949-957.

[35] RAHMOUNI, M., & MBARKI, S., "An End-to-End Code Generation from UML Diagrams to MVC2 Web Applications". International Review in Computers and Software (IRECOS), Vol.8, No.9, 2013, pp.2123-2135.

[36] RAHMOUNI, M., & MBARKI, S., "MDA-Based Modeling and Transformation to Generate N-Tiers Web Model". Journal Of Software (JSW), Vol.10, No.3, 2015, pp. 222-238, DOI: 10.17706/jsw.10.3.222-238.

AUTHOR PROFILE

M'hamed RAHMOUNI received the Diploma of Higher Approfondie Studies in Computer Science and Telecommunication from the faculty of science, Ibn Tofail University, Morocco, 2007, and Doctorat of High Graduate Studies degrees in Computer Sciences from Ibn Tofail University, Morocco, January 10, 2015 . He participated in various international congresses in MDA (Model Driven Architecture) integrating new technologies XML, EJB, MVC, Web Services, etc. and he published many papers in the MDA domain.

Samir MBARKI received the B.S. degree in applied mathematics from Mohammed V University, Morocco, 1992, and Doctorat of High Graduate Studies degrees in Computer Sciences from Mohammed V University, Morocco, 1997. In 1995 he joined the faculty of science Ibn Tofail University, Morocco where he is currently a Professor in Department of computer science. His research interests include software engineering, model driven architecture, software metrics and software tests. He obtained an HDR in computer Science from Ibn Tofail University in 2010